Vertica Knowledge Base Article

Copy and Restore Data from a Vertica Cluster to a Backup

Document Release Date: 10/25/2018

Contents

Copy and Restore Data from a Vertica Cluster to a Backup

This is the first document in a three-part series on backup and restore, see below for parts 2 and 3:

Part 2: Copying Data Between Two Similar Vertica Clusters

Part 3: Copying Data Between Two Dissimilar Vertica Clusters

Copy and Restore Data from a Vertica Cluster to a Backup: PDF Format

Part 1: Copy and Restore Data from a Vertica Cluster to a Backup

You can create a backup and restore data between a Vertica cluster and a backup location, with these options:

- Full Restore from a Full Backup—Create a backup of your entire Vertica database and restore your entire Vertica database.
- Restore Specific Tables from a Full Backup—Create a backup of your entire Vertica database and restore only specific tables.
- Restore Tables from a Partial Backup—Create a backup of specific tables and restore the specific tables.
- Side-by-Side Table Restore—Create a backup of original tables and current tables to compare and reconcile the original and the restored tables.

For the purpose of demonstration and conciseness, the example in this document uses a 2-node cluster, but a 3-node cluster with a K-safety level of 1.

Types of Backup

The Vertica `vbr` utility supports the following types of backup:

Type of Backup	Limitations
Full backup and restore	A full backup is a complete copy of the database catalog, its schemas, tables, and other objects. This type of backup provides a complete image of the database at the time the backup occurred.
Object-level backup and	An object-level backup consists of one or more schemas or tables, or a group of such objects. The parts of the object-level backup do not contain the entire

restore	database.
Hard-link local backup and restore	A hard-link local backup can be a full or object-level backup. The backup consists of the complete copy of the database catalog and a set of hard file links to the corresponding data files.

For more information, see Types of Backup and VBR Utility Reference in the Vertica documentation.

Source and Target Clusters

For the purpose of this document:

- The primary cluster is referred to as the *source*.
- The secondary cluster is referred to as the *target*.
- Backup always occurs on the source.
- Restore always occurs on the target.
- For backup the cluster is referred to as the source cluster.
- For restore the cluster is referred to as the target cluster.
- The IP addresses for the source cluster nodes are 10.100.0.66 and 10.100.0.77.
- The IP addresses for the target cluster nodes are 10.100.0.88 and 10.100.0.99.

Initialize a Backup Configuration File

As of Vertica 7.2, you must initialize your backup location. The configuration file includes metadata for the database snapshots and the backup files. Vertica uses this configuration file for incremental backup.

To initialize a backup configuration file, follow these steps:

1. On the source cluster, create a backup configuration file `backup_snapshot`.

```
$ /opt/vertica/bin/vbr.py --setupconfig
Snapshot name (backup_snapshot): backup_snapshot
Number of restore points (1): 5
Specify objects (no default):
Object restore mode (coexist, createOrReplace or create)
(createOrReplace): createOrReplace
Vertica user name (dbadmin): mydatabase
Node v_ mydatabase_node0001
Backup host name (no default): host1
Backup directory (no default): backup_location
Node v_ mydatabase_node0002
Backup host name (no default): host2
Backup directory (no default): backup_location
Config file name (backup_snapshot.ini): backup_snapshot
Saved vbr config to backup_snapshot.
```

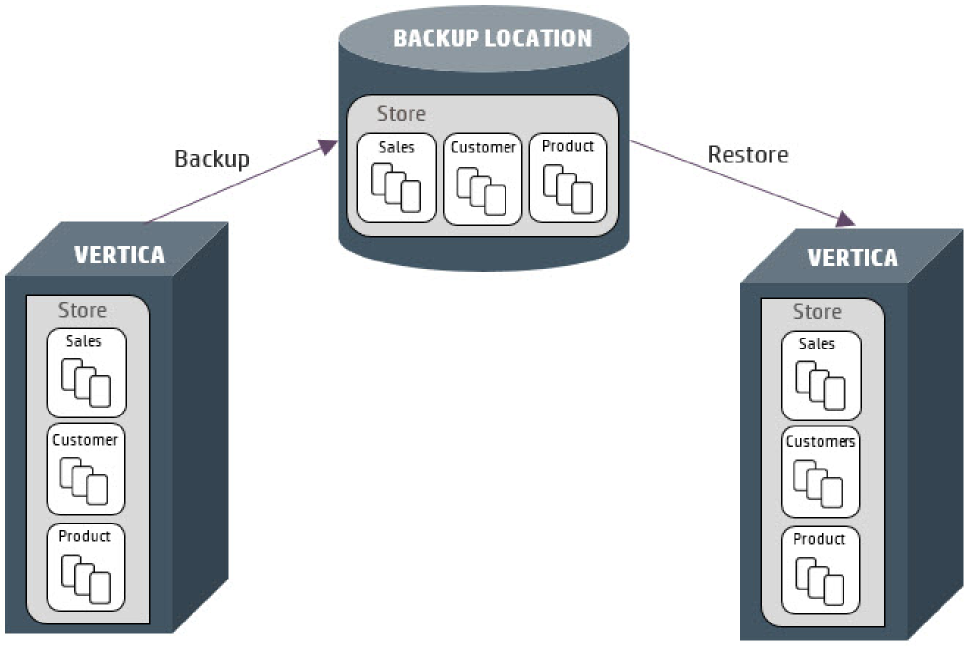
2. On the source cluster, initialize your backup location. To initialize your backup location, run the `init` statement.

```
$ /opt/vertica/bin/vbr.py -t init --config-file backup_snapshot.ini
Initializing backup locations.
Backup locations initialized.
```

Full Restore from a Full Backup

Use full database backup in the following scenarios:

- Your geographical location is hit by a disaster that results in total loss of the cluster.
- Your database experiences multi-node, unrecoverable corruption that results in total loss of data.



Prerequisites:

- The target cluster must have the same number of nodes as the source cluster.
- The target cluster must have the same IP address, dbadmin user, and database name as the source cluster.
- The source and the target cluster must have the same Vertica version.

- You must have already created a full backup of your database with the backup configuration file, at a backup location.

How to Perform a Full Backup and Restore

Before you can perform a full restore of your data, you must have created a full backup of your database. Below is an example of a making a backup on a two node cluster.

```
$ /opt/vertica/bin/vbr.py -t backup --config-file backup_snapshot.ini
Starting backup of database mydatabase.
Participating nodes: v_mydatabase_node0001, v_mydatabase_node0002.
Snapshotting database.
Snapshot complete.
Approximate bytes to copy: 348431103 of 348431103 total.
Copying backup metadata.
Finalizing backup.
Backup complete!
```

> Note If you back up the database at different times, Vertica records the differences between each backup. Each point in time backup is referred to as a restore point.

To perform a full restore, follow these steps:

1. On the target cluster, create an empty database with same number of nodes, node name, database name, dbadmin user, and IP addresses.

```
$ admintools -t create_db -d mydatabase -s v_mydatabase_node0001,v_
mydatabase_node0002
$ admintools -t stop_db -d mydatabase
```

2. On the target cluster, create a full restore of the database you backed up.

```
$ /opt/vertica/bin/vbr.py -t restore --archive 20150914_150023 --config-
file backup_snapshot.ini
Starting full restore of database mydatabase.
Participating nodes: v_mydatabase_node0001, v_mydatabase_node0002.
Restoring from restore point: backup_snapshot_20150914_150023
Syncing data from backup to cluster nodes.
Restoring catalog.
Restore complete!
```

> Note You can restore to a specific point in time (restore point) using the `–archive` argument as shown above.

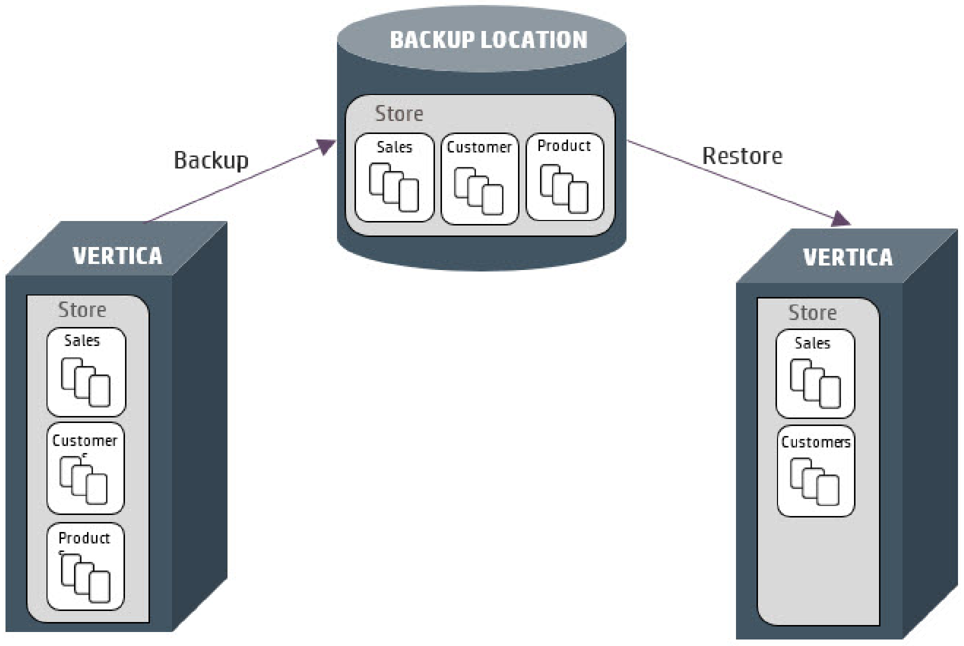
Advantages and Limitations of a Full Backup and Full Restore

Advantages	Limitations
• The database can be backed up while running. • Multiple incremental backups possible. • Can recover from complete a database loss. • Fast incremental backups at regular intervals, if no significant data changes.	• Backup takes a U lock on every table, preventing the Tuple Mover from deleting old containers. • To restore the target database must be down. • To restore the database, source and target clusters must have the same: • Number of nodes • IP addresses • Database name and dbadmin users • Vertica version

For more information, see Creating Full and Incremental Backups in the Vertica documentation.

Restore Specific Tables from a Full Backup

Suppose you accidentally truncate a table. Recovering the entire database might result in losing any data that you added after the last backup. In such situations, restoring specific tables is appropriate.



Prerequisites:

- The target cluster must have the same number of nodes as the backup cluster.
- The target cluster must have the same IP address, dbadmin user, and database name as the source cluster.
- You must have already created a full backup of your database with the backup configuration file, at a backup location.

How to Perform a Single Table Restore from a Full Backup

As shown in the previous section, you must have already created a full backup of your database. Suppose, sometime after the full backup, you accidentally truncate the `store_sales` table.

```
=> SELECT COUNT(*) FROM STORE.STORE_SALES;
COUNT
---------
5000000

=> TRUNCATE TABLE STORE.STORE_SALES;
TRUNCATE TABLE
```

Next, assume that you added data or tables after the full backup. For the purpose of this example, let’s create a new table, named `table_order`.

```
=> CREATE TABLE TABLE_ORDER (I INT);
CREATE TABLE

=> INSERT INTO TABLE_ORDER VALUES (1);
OUTPUT
--------
1

=> COMMIT DURABLE;
COMMIT
```

If you do a full restore to the point of your last full backup, you lose new data, which is `table_order`. You want to restore *only* the truncated table rather than the whole database. In such a situation, you perform a table restore to restore only the truncated table. By doing so, you do not lose the newly added data.

This example shows how this might work:

On the source cluster, restore only the truncated store_sales table from the full backup.

```
$ /opt/vertica/bin/vbr.py -t restore --archive 20150914_195908 --config-
```

```
file backup_snapshot.ini --restore-objects "store.store_sales"
Starting object restore of database mydatabase.
Participating nodes: v_mydatabase_node0001,v_mydatabase_node0002.
Objects to restore: store.store_sales.
Restoring from restore point: backup_snapshot_20150914_195908
Loading snapshot catalog from backup.
Extracting objects from catalog.
Syncing data from backup to cluster nodes.
Finalizing restore.
Restore complete!
```

You have restored the `store_sales` table from your last full backup. The object-level restore did not delete the `table_order`.

```
=> SELECT COUNT(*) FROM STORE.STORE_SALES;
COUNT
---------
5000000

The table_order is unaffected
=> SELECT * FROM TABLE_ORDER;
I
---
1
```

> Note In some cases, the table you are restoring has primary or foreign key constraint. To maintain referential integrity make sure you restore the tables that link the constraints.

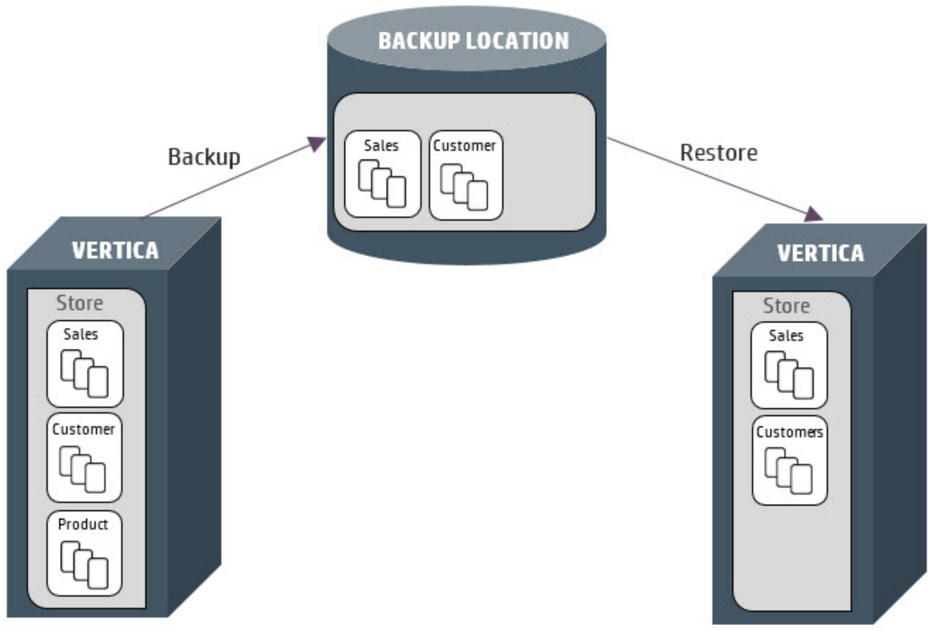
Advantages and Limitations of Full Backup and Partial Restore

Advantages	Limitations
• Multiple incremental backups possible. • Restore specific tables. • Can restore a specific table while the database is up and running. • Incremental backups at regular intervals possible, if no significant data change. • Faster than full database restore.	• To perform restore, source and target clusters must have the same: • Number of nodes • IP addresses • Database name and dbadmin users • Vertica version

For more information, see Creating Object-Level Backups in the Vertica documentation.

Restore Tables from a Partial Backup

Assume that your database consists of multiple tables whose data does not change, such as *Products*. However, other tables, like *Sales*, change frequently. Partial backup and restore allows you to back up and restore *only* specific tables of your database.



Prerequisites:

- The target cluster must have the same number of nodes as the backup cluster.
- The target cluster must have the same IP address, dbadmin user, and the database name as the source cluster.
- You must have already created a full backup of your database with the backup configuration file, at a backup location.

How to Perform Specific Table Restore from a Partial Backup

You can back up specific tables and restore a specific subset of tables as follows:

Back Up Specific Tables

1. On the source cluster, specify the tables for a backup. This example shows how to create a backup of tables, `store.store_sales` and `public.new_table`.

```
$ cat objectbak.ini
[Misc]
snapshotName = objectbak
restorePointLimit = 5
objects = store.store_sales,public.new_table
```

```
objectRestoreMode = createOrReplace

[Database]
dbName = mydatabase
dbUser = dbadmin
dbPromptForPassword = False

[Transmission]

[Mapping]
v_mydatabase_node0001 = host1:/vertica/data/objbackup
v_mydatabase_node0002 = host1:/vertica/data/objbackup
```

2. On the source cluster, perform the backup of the two tables.

```
$ /opt/vertica/bin/vbr.py -t backup --config-file objectbak.ini
Starting backup of database mydatabase.
Objects: ['public.new_table', 'store.store_sales']
Participating nodes: v_mydatabase_node0001, v_mydatabase_node0002.
Snapshotting database.
Snapshot complete.
Approximate bytes to copy: 51 of 108630328 total.
Copying backup metadata.
Finalizing backup.
Backup complete!
```

Restore a Specific Table

If you have a backup of specific tables of the database, you can perform a partial restore. This example shows how to restore *only* the `store.store_sales` table on the source cluster.

```
$ /opt/vertica/bin/vbr.py -t restore --config-file objectbak.ini --
restore-objects "store.store_sales"
Starting object restore of database mydatabase.
Participating nodes: v_mydatabase_node0001,v_mydatabase_node0002.
Objects to restore: store.store_sales.
Restoring from restore point: objectbak_20150914_203903
Loading snapshot catalog from backup.
Extracting objects from catalog.
Syncing data from backup to cluster nodes.
Finalizing restore.
Restore complete!
```

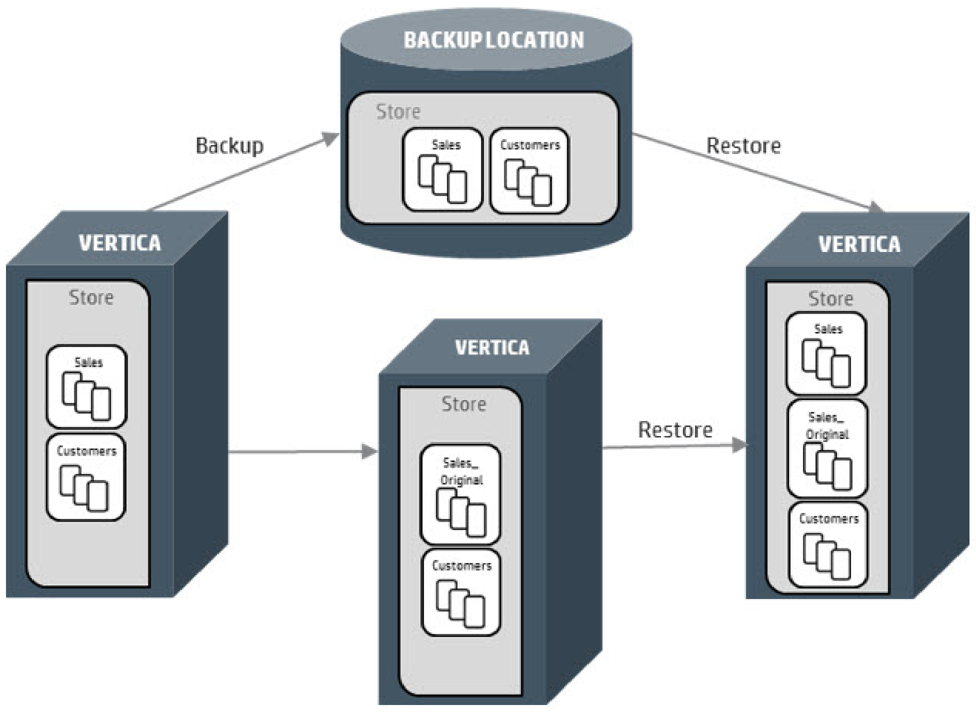
Advantages and Limitations of Restoring Tables from a Partial Backup

Advantages	Limitations
• Back up possible when the database is up and running. • Incremental backups at regular intervals possible, if no significant changes. • Restore specific tables. • Faster than full backup. • Saves disk space at backup location. • Data from other tables not lost during restoration	• To perform restore, source and target clusters must have the same: • Number of nodes • IP addresses • Database name and dbadmin users • Vertica version • This operation is only available for tables and partitions.

For more information, see Restoring Object-Level Backups in the Vertica documentation.

Side-by-Side Table Restore

Assume that on a Monday morning you learn that of the jobs that ran over the weekend, 9 passed and 1 failed. The failed job has modified data in an unpredictable way. You now want to compare your existing data with a backup of Friday's data to determine the differences.



Prerequisites:

- The target cluster must have the same number of nodes as the backup cluster.
- The target cluster must have the same IP address, dbadmin user, and the database name as the source cluster.
- You must have already created a full backup of your database with the backup configuration file, at a backup location.

How to Perform Side-by-Side Table Restore

This example shows how you can rename the existing tables and projections and restore the original database. Renaming the table allows you to compare and reconcile the original and the restored tables.

To perform a side-by-side table restore, follow these steps:

1. Rename the existing `store_sales` table.

```
=> ALTER TABLE STORE.STORE_SALES RENAME TO STORE_SALES_ORIGINAL;
ALTER TABLE
=> ALTER PROJECTION STORE.STORE_SALES_SUPER RENAME TO STORE_SALES_
ORIGINAL_SUPER;
ALTER PROJECTION
```

2. On the source cluster, restore the store_sales table from a backup.

```
$ /opt/vertica/bin/vbr.py -t restore --config-file objectbak.ini --
restore-objects "store.store_sales"
Starting object restore of database mydatabase.
Participating nodes: v_mydatabase_node0001,v_mydatabase_node0002.
Objects to restore: store.store_sales.
Restoring from restore point: objectbak_20150914_203903
Loading snapshot catalog from backup.
Extracting objects from catalog.
Syncing data from backup to cluster nodes.
Finalizing restore.
Restore complete!
```

   After the preceding command, both `store_sales_original` and the `store_sales` backup exist.

3. Now, you can inspect the data as it was before and after the failed job. You can compare the two tables and update the restored store_sales table with the correct data. Upon comparison, you can reconcile the `store_sales_original` with the `store_sales` table.
   Swap the partitions between the original table `store_sales_original` and the restored table `store_sales` as needed.

```
=> DT STORE_SALES*
List of tables
Schema | Name                  | Kind  | Owner
-------+-----------------------+-------+--------
store  | store_sales           | table | dbadmin
store  | store_sales_original  | table | dbadmin

=> SELECT SWAP_PARTITIONS_BETWEEN_TABLES ('STORE.STORE_SALES', '1', '1',
'STORE.STORE_SALES_ORIGINAL');
```

Advantages and Limitations of Side-by-Side Backup and Restore

Advantages	Limitations
• Retrieve backups of existing tables. • Perform operations between live and backed-up tables. • After backup, both the original and the restored tables available.	• To restore the database, source and target clusters must have the same: • Number of nodes • IP addresses • Database name and dbadmin users • Vertica version • Available only for tables and partitions.

For More Information

For More Information About…	See…
Vertica Community Edition	https://my.vertica.com/community/
Vertica Documentation	http://my.vertica.com/docs/latest/HTML/index.htm
Big Data and Analytics Community	https://my.vertica.com/big-data-analytics-community-content/

Copy and Restore Data from a Vertica Cluster to a Backup